COMP90042

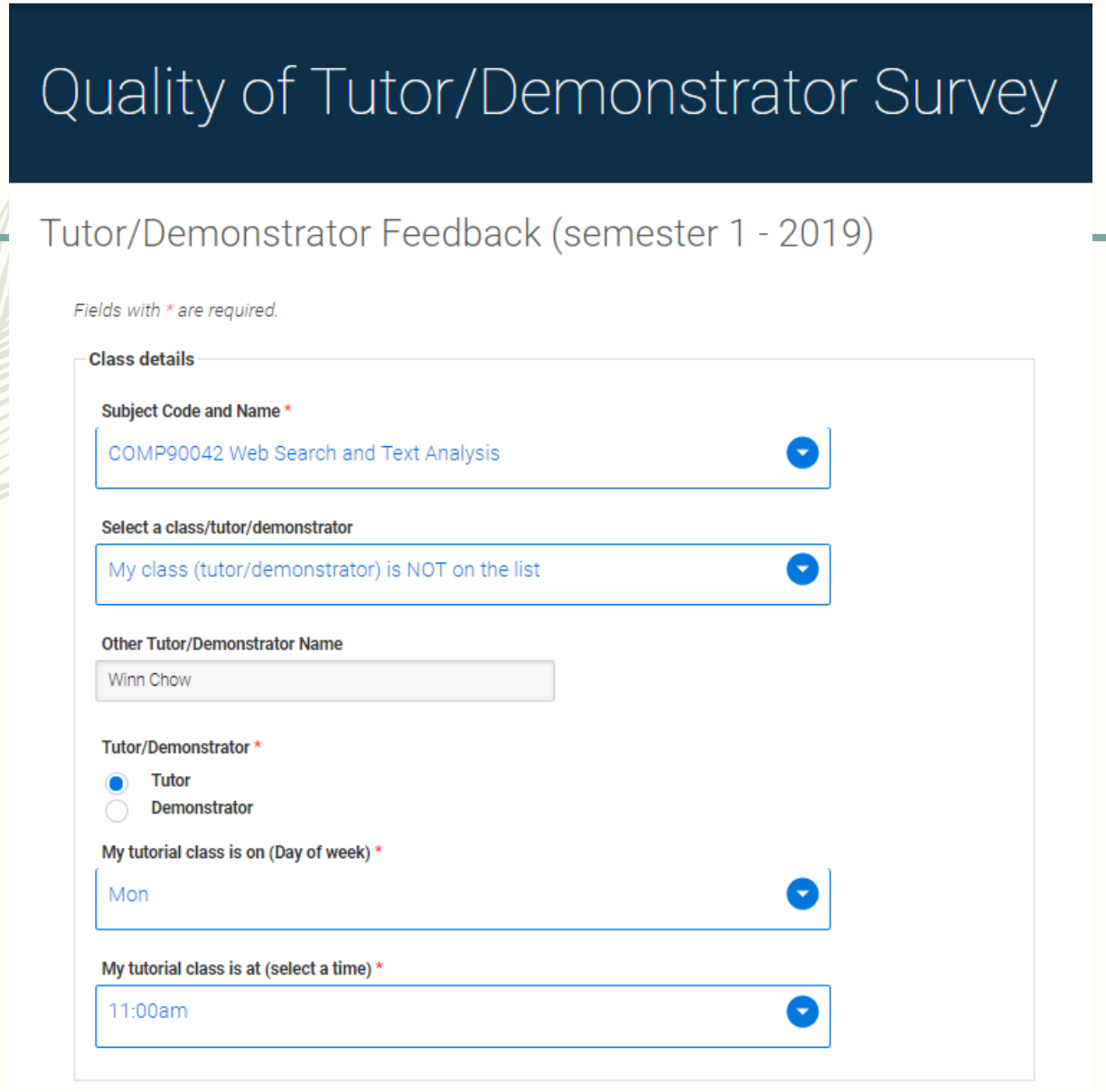
Web search and text analysis

Workshop Week 11

Your tutor

- Winn Chow (Senior Tutor)
- winn.chow1@unimelb.edu.au
- Office: Doug McDonell - 9.23
- Here, you can find my workshop slides:
- https://github.com/winnchow/COMP90042-Workshops

https://apps.eng.unimelb.edu.au/casmas/index.php?r=qoct/subjects



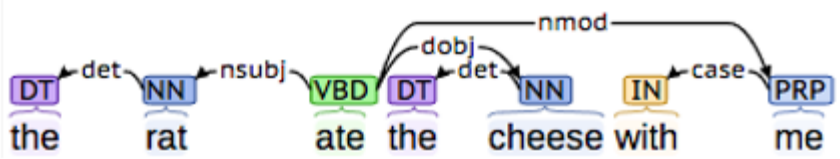
Q1 and Q2

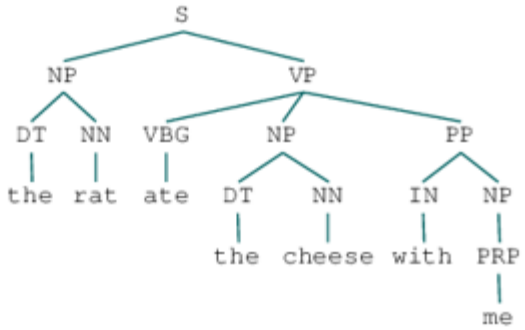
1. Using typical dependency types, construct (by hand) a dependency parse for the following sentence: *Yesterday, I shot an elephant in my pyjamas.* Check your work against the output of the online GUI for the Stanford Parser (`http://nlp.stanford.edu:8080/parser/index.jsp`).

2. In what ways is (transition–based, probabilistic) dependency parsing similar to (probabilistic) CYK parsing? In what ways is it different?

Why dependencies?

- Dependency tree more directly represents the core of the sentence: *who did what to whom?*
  - captured by the links incident on verb nodes, e.g., NSUBJ, DOBJ etc; easier to answer questions like:
    - what was the main thing being expressed in the sentence (*eating* = root)





  - more minor details are buried deeper in the tree (e.g., adjectives, determiners etc)

Buffer	Stack	Action
Yesterday I shot an elephant in my pyjamas		Shift
I shot an elephant in my pyjamas	Yesterday	

Buffer	Stack	Action
Yesterday I shot an elephant in my pyjamas		Shift
I shot an elephant in my pyjamas	Yesterday	Shift
shot an elephant in my pyjamas	Yesterday, I	Shift
an elephant in my pyjamas	Yesterday, I, shot	Arc-Left (I <- shot)
an elephant in my pyjamas	Yesterday, shot	Arc-Left (Yesterday <- shot)
an elephant in my pyjamas	shot	

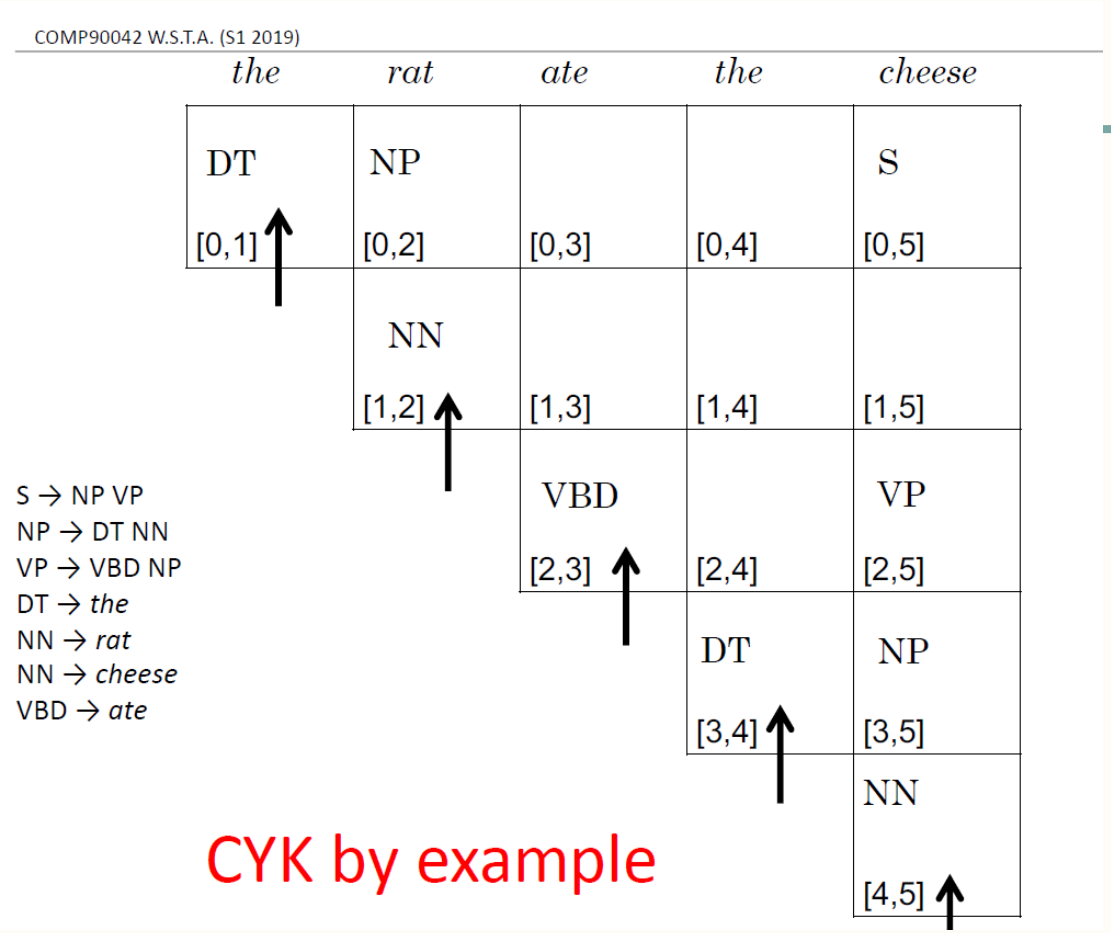
Stanford Parser

Universal dependencies

```
nmod:tmod(shot-4, Yesterday-1)
nsubj(shot-4, I-3)
root(ROOT-0, shot-4)
det(elephant-6, an-5)
dobj(shot-4, elephant-6)
case(pyjamas-9, in-7)
nmod:poss(pyjamas-9, my-8)
nmod(shot-4, pyjamas-9)
```

Buffer	Stack	Action
Yesterday I shot an elephant in my pyjamas		Shift
I shot an elephant in my pyjamas	Yesterday	Shift
shot an elephant in my pyjamas	Yesterday, I	Shift
an elephant in my pyjamas	Yesterday, I, shot	Arc-Left (I <- shot)
an elephant in my pyjamas	Yesterday, shot	Arc-Left (Yesterday <- shot)
an elephant in my pyjamas	shot	Shift
elephant in my pyjamas	shot, an	Shift
in my pyjamas	shot, an, elephant	Arc-Left (an <- elephant)
in my pyjamas	shot, elephant	Arc-Right (shot -> elephant)
in my pyjamas	shot	Shift
my pyjamas	shot, in	Shift
pyjamas	shot, in, my	Shift
	shot, in, my, pyjamas	Arc-Left (my <- pyjamas)
	shot, in, pyjamas	Arc-Left (in <- pyjamas)
	shot, pyjamas	Arc-Right (shot -> pyjamas)
	shot	<done>

CYK


COMP90042 W.S.T.A. (S1 2019)
the
rat
ate
the
cheese
DT
[0,1]
NP
[0,2]
[0,3]
[0,4]
S
[0,5]
NN
[1,2]
[1,3]
[1,4]
[1,5]
VBD
[2,3]
[2,4]
VP
[2,5]
DT
[3,4]
NP
[3,5]
NN
[4,5]
S → NP VP
NP → DT NN
VP → VBD NP
DT → the
NN → rat
NN → cheese
VBD → ate
CYK by example

Q2

- Both methods are attempting to determine the structure of a sentence; both methods are attempting to disambiguate amongst the (perhaps many) possible structures licensed by the grammar by using a probabilistic grammar to determine the most probable structure.
- Both methods process the tokens in the sentence one–by–one, left–to–right.

Q2

- Although POS tags are implicitly used in constructing the "oracle" (training), the depedency parser doesn't explicitly tag the sentence.
- The transition–based dependency parser can potentially take into account other (non–local) relations in the sentence, whereas CYK's probabilities depend only on the (local) sub-tree.
- CYK adds numerous fragments to the chart, which don't end up getting used in the final parse structure, whereas the transition–based dependency parser only adds edges that will be in the final structure.

Q3 and Q4

3. What is **Discourse Segmentation**? What do the segments consist of, and what are some methods we can use to find them?
4. What is an **anaphor**?
   (a) What is **anaphora resolution** and why is it difficult?
   (b) What are some useful heuristics (or features) to help resolve anaphora?

Beyond the sentence

- **Discourse**: a coherent, structured group of sentences (utterances)

Yesterday, Ted was late for work. [It all started when his car wouldn't start. He first tried to jump start it with a neighbour's help, but that didn't work.] [So he decided to take public transit. He walked 15 minutes to the tram stop. Then he waited for another 20 minutes, but the tram didn't come. The tram drivers were on strike that morning.] [So he walked home and got his bike out of the garage. He started riding but quickly discovered he had a flat tire. He walked his bike back home. He looked around but his wife had cleaned the garage and he couldn't find the bike pump.] He started walking, and didn't arrive until lunchtime.

Q3

- By interpretting the task as a boundary–finding problem, we can use rule–based or unsupervised methods to find sentences with little lexical overlap (suggesting a discourse boundary). We can also use supervised methods, by training a classifier around paragraph boundaries.

Anaphors

- **Anaphor**: linguistic expressions that refer back to earlier elements in the text
- Anaphors have a **antecedent** in the discourse, often but not always a noun phrase

Yesterday, Ted was late for work. It all started when his car wouldn't start.

- Pronouns are the most common anaphor
- But there are various others
  - Demonstratives (*that problem*)
  - Definites (*the problem*)

Q4a

- This is the problem of working out which element (generally a noun or noun phrase, but sometimes a whole clause) a given anaphor is actually referring to.
- For example:

  Mary gave John a cat for **his** birthday. (i) **She** is generous. (ii) **He** was surprised. (iii) **He** is fluffy.

  his [birthday] obviously refers to John; (i) (presumably) refers to *Mary*; (ii) (presumably) refers to *John*; and (iii) (presumably) refers to *[the] cat*.

Q4b

- The most obvious (but inherent unreliable) heuristic is the **recency heuristic**: given multiple possible referents (that are consistent in meaning with the anaphor), the mostly intended one is the one most recently used in the text.
- A better heuristic is that the most likely referent (consistent in meaning with the anaphor) is the focus of the discourse (the "center").
- We can also build a supervised machine learning model, usually based around the semantic properties of the anaphor/nearby words and the sentence/discourse structure.